

## Patent Application Specification Page

### 1. Descriptive title of Invention:

Computers that communicate in the English Language and complete work assignments by reading English Language sentences

### 2. Cross Reference to Related Applications

Patent Number 6,101,490

Others may apply

### 3. Statement Regarding Fed sponsored R & D

None

### 4. Computer program modules that make up the computer application:

```
% - EnglishWorks (tm),
%   Copyright 1987 - 2001
% - by Charles M. Hatton
% - Sections of this program covered by U.S. Patent 6,101,490
% - Program name: ewloader.pl

%*****
% - WARNING WARNING: YOU MUST USE ensure_loaded/1 in order
%   to build a application.
% - All path pointer definitions are now done in brain1.pl

:- ensure_loaded('a:\Ews\ewsu1.pl').
:- ensure_loaded('a:\Ews\retract1.pl').
:- ensure_loaded('a:\Ews\dic0.pl').
:- ensure_loaded('a:\Ews\dic00a.pl').
:- ensure_loaded('a:\Ews\dic1a.pl').
:- ensure_loaded('a:\Ews\dic2.pl').
:- ensure_loaded('a:\Ews\dmempl1.pl').
:- ensure_loaded('a:\Ews\ewsmenu1.pl').
:- ensure_loaded('a:\Ews\keyword1.pl').
:- ensure_loaded('a:\Ews\Locator1.pl').
:- ensure_loaded('a:\Ews\logic1.pl').
:- ensure_loaded('a:\Ews\liver1.pl').
```

```
:- ensure_loaded('a:\Ews\liver2.pl').
:- ensure_loaded('a:\Ews\mempl.pl').
:- ensure_loaded('a:\Ews\memutl1.pl').
:- ensure_loaded('a:\Ews\parsenp1.pl').
:- ensure_loaded('a:\Ews\parsevp1.pl').
:- ensure_loaded('a:\Ews\semant1.pl').
:- ensure_loaded('a:\Ews\semant2.pl').
:- ensure_loaded('a:\Ews\sql.pl').
:- ensure_loaded('a:\Ews\syn2r0.pl').
:- ensure_loaded('a:\Ews\syn2r1.pl').
:- ensure_loaded('a:\Ews\syn2r2.pl').
:- ensure_loaded('a:\Ews\syn2r3.pl').
:- ensure_loaded('a:\Ews\synmst1.pl').
:- ensure_loaded('a:\Ews\synmst2.pl').
:- ensure_loaded('a:\Ews\syntax2.pl').
:- ensure_loaded('a:\Ews\syntax3.pl').
:- ensure_loaded('a:\Ews\whitecl.pl').
:- ensure_loaded('a:\Ews\readdisk1.pl').
:- ensure_loaded('a:\Ews\brain1.pl').
```

Several other software modules are used in this application including the dictionary systems, computer memories, initialization files, start up files, APIs, DLLs, SQL databases, text files.

#### 5. Background of Invention:

Work started in 1987, follow by US Patent filing in 1991, then re-filed in 1993, then filed under appeal, U.S. Patent awarded 8/2000 (6,101,490) others pending. Current patent application (this one) makes claims regarding computers that communicate with each other in the English Language. Current computer application has extensive functionality added in order to allow computers to communicate with each other and people in the English Language.

#### 6. Brief Summary of Invention:

Computer application is trained by storing English Language declarative sentences in N number of user-defined memories. Input system gets English Language sentence from M devices and computer application applies O number of sentences to memories. If a match is found, attached actions to stored English Language sentences cause computer to execute that activity. Attached actions can be other English Language sentences that are fed back to input (recursive) until no more sentences can be processed. SQL databases, text files and other sources (including other computers) can input English Language sentences to application to cause work activities to be completed by computer application. Multiple computer applications can run under the same computer and communicate in English or other computer across the network.

✓  
5

#### 7. Brief Description of Drawings:

See Figure 1 – Drawing described by item 6 above.

8. Claims

9. Abstract

10. Disclosure

11. Figures  
12. References  
13. Definitions  
14. Summary  
15. Detailed Description  
16. Claims  
17. Abstract  
18. Disclosure

Patent application for: “Computers that communicate in the English Language and complete work assignments by reading English Language sentences.”

Claims:

1. Said application can carry out spontaneous (positive value work occurs) English Language conversations with other said application(s) running on various computers and devices.

2. Computer memory for said application is English Language sentences that may be constructed by people or machine using the English Language.

3. Said application can dynamically change the way it processes an English Language sentence using said previous English Language sentences which reconfigures said application.

4. Said application can dynamically switch memories (defined as a group of English Language sentences) based on an English Language sentence input from another device, human, or said application sending a new English Language sentence fed back into the input. in Figure 1 item (13) from said application’s memories in Figure 1 items (5), (8), (11). This occurs because sentences stored in memory can have any number of attached sentences that can be sent back into the said applications input or to other devices in Figure 1 item (14). Said other devices can send English

Language sentences back to first said application in response to said applications sentences over a network.

5. Said application can be programmed by storing ordinary English Language sentences.

6. Said application can read English Language sentences stored in text files in order to teach itself (said application) with English Language sentences so that after said application has trained itself, a said last sentence in the text file that has trained said application can be used to cause said application to respond correctly to what said application has learned.

7. Said application can take an input English Language sentence, find a fuzzy match in 1 of N English Language memories (made up of English Language sentences) and having attached English Language sentences that are fed back to the input of said application in Figure 1 item (13) such that said application utilizes several memories in Figure 1 items (5), (8), and (11) so that said application carries out a number of coordinated actives to complete a total task made by calling one or more English Language sentences connected to 1 of N English Language sentences in Figure 1 items (5) (8), and (11).

8. Said application with machine or human English Language input can find an existing English Language sentence using fuzzy logic with N number of actions or attached

English Language sentences that can be fed back into the input at Figure 1 item (13) or go to the output in Figure 1 item(14) such that an English Language sentence is sent to another said application operating on other computers or devices such that said device replies in response with said English Language sentence originating from said first application. Said original application would complete English Language conversation.

9. Said application can change entire memory systems based on an English Language sentence.

10. Said application is context sensitive dynamically switching memories (made up of English Language sentences) based on input sentences.

11. Said application can process multiple English Language sentences sent to the said application as a text file, a tape recording, human, English Language sentences synthesized by a computer or any said device capable of transmitting English Language sentences.

12. Said application can logically process multiple input English Language sentences as in: please play a card game. If you can play cards then show the card game instructions. If you can not play cards, play chess.

13. Said application can take two identical sentences and return a difference answer by telling said application to switch memories (context).

14. Said application dynamically takes a single sentence and builds several sentences to search text or other file types. In this example, the following input sentence: "Who is the mayor of Chicago?" causes the said application to build two sentences: "Who is the mayor?" and "Who is the mayor of Chicago?" The first sentence: "Who is the mayor?" causes the said application to switch to mayor memory. Mayor memory then opens the appropriate text memory that when the said application receives the second sentence: "Who is the mayor of Chicago?" said application displays the appropriate answer.

15. Any device that can process English Language like constructs, parsing components into actions (verbs), prepositional phrases, noun phrases, verb phrases, adverbs, and other components such that said parsing can communicate with said mechanisms of said patent application.

16. Tagging any electronic transaction with multiple sentences such that the transaction becomes data and the tagged multiple sentences attached to the transaction(s) are English Language instructions telling said receiver or other devices of information what should be done.

17. Using English Language tags in the form of English Language sentences or English

Language words to be used to define the likeness of data so that said data can be integrated in order to build new data with added value.

18. Defining any suitable standard or non standard database system holding English

Language sentences such that said English Language sentences can be used to re-program said computer application.

19. Using said computer program to translate English Language sentences made from

said imaging device so that said application can respond to English Language sentences made by vision devices. Said devices can be any device capable of translating observation of images, voice, and other transducers into English Language sentences and components.

20. An English Language declarative sentence (noun phrase plus a verb phrase) is

reformatted such that its verb phrase (made up of a verb plus a noun phrase) where the verb phrases noun phrase may be substituted with multiple sentences such as: computer play cards is “go to game memory. please play a game of cards.” Where: is “go to game memory. please play a game of cards.” is the verb phrase and its noun phrase is “go to game memory. please play a game of cards.”

21. Claim 21 is related to claim 20 in that the back part of any sentence can call the front

part of any other sentence stored in the same memory or in other memories. The



same is true for native databases, text databases, SQL databases, and any other  
databases that store human language in any medium.

22. The computer program uses sentence logic as in: “run a computer card game if:  
switch to your science memory. Why is the sky blue? Otherwise call your doctor.

23. If the computer program can not answer the question: Why is the sky blue? it will be  
forced to learn the answer to this question by storing the appropriate English  
Language sentence in one of its English Language memories.

24. The computer program may learn by sending one or more English Language  
sentences to the input Figure 1 item 1 or from English Language sentences read from  
memory at Figure 1 item 13 coming from Figure 1 item 12.

25. An English Language sentence may be sent to the input Figure 1 items 1 instructing  
said computer application to read English Language sentences from any media such  
that when computer applications reads English Language sentences (Figure 3) it has  
learned something by storing a sequence of English Language sentences in said  
computer application memory at Figure 1 items 5, 8, 11.

26. Said computer application may learn by an inference method described in US Patent  
(6,101,490) such that inference causes new English Language sentence to be stored  
in said computer application such that when computer is told: “My car will not start.

What should I do?” said computer makes an inference from “cars transport people” to “taxi transport people” and stores said sentence in memory at Figure 1 items 5, 8, or 11 and answers said question “What should I do? by telling said user to “take a taxi.” Said inference “taxi transport people” is stored in the appropriate computer memory at Figure 1, items 5, 8, or 11.

27. Said computer memories may be any media that can be accessed via computers in a networked or non networked environment.

28. Said computer application may gain new knowledge by storing English Language sentences that may be converted from or to other computer data formats.

29. Said computer application may be told in English Language sentences to add new memory by a copy and past method where by new sections of memory are added to computers said memory system.

30. Said computer application uses said sentence logic as in: “run the computer game of solitaire if: locate the solitaire card game. Otherwise learn to run the game of solitaire.” Said computer application uses said logic to modify said computer memory. Said sentences (defined in this claim) may come from Figure 1 items 1, 13.

158 31. Said computer application may generate goals in the form of English Language  
159 sentences such that said computer program may evolve said English Language  
160 memories at Figure 1 items 5, 8, and 11 where items 5, 8, and 11 could be remote  
161 memory via a computer network or some other method to connect to memory that  
162 said computer application would evolve its memory by a trial and error method to  
163 eventually solve said English Language goal.

164

165

166 Abstract1:

167 This patent application defines computer processes that support using the English  
168 language as a computer language. Specifically, methods are defined that let the  
169 computer: 1) process input English Language sentences from its memory which are also  
170 made up of English Language sentences. 2) use fuzzy logic to allow the computer to  
171 respond to differently worded sentences to arrive at the same action i.e., play a card  
172 game. Run solitaire. Where (play a game.) also plays the computer game of solitaire; 3)  
173 define contexts so that the computers can do: Go to New York. Who is Jim Smith? Go to  
174 Washington. Who is Jim Smith? Where answering the question, Who is Jim Smith?  
175 depends on the context: Go to New York or Go to Washington; 4) respond to multiple  
176 input sentences such as: What is the time. Get my word processor. Go to Greenburg. Who  
177 is Rod Smith; 5) where each sentence can be connected to other sentences so that in item  
178 4) Get my word processor is connected to: Go to document memory. Get word; 6)  
179 process numbers in sentences to be variables or numbers i.e. play the game of 1.222 and  
180 What is 3.44 times 22.44; 7) use multi sentence so that one sentence preceding another

181 can change the processing methods for the successor sentence such as: show to 2 decimal  
182 places. What is 3.3333 times 2.44467? – shows answer as 8.14; 8) build the computer's  
183 memory system from English language text files; 9) talk to each other in English so they  
184 can perform parallel computing; 10) use transducers to change voice, vision, touch, other  
185 signals to English language so the application can process those signals in its English  
186 language memory systems; 11) use any device to convert to the English language; 12) do  
187 inference or common sense reasoning to do the following: My car will not start. What  
188 should I do?; 13) use deductive based reasoning to do: My car will not start. What is  
189 wrong?; 14) integrate inference based reasoning with deductive based reasoning; 15)  
190 Make the computer read English language text files and store those text files in one of its  
191 memory systems; 16) make the text file that trained the application in (15) ask the  
192 application what it has learning to test that learning; 16) integrate English language data  
193 sets associated by sentence memory i.e. Go to English Works memory. Get the  
194 enhancement list. If not the enhancement list, then get design notes; 17) be a software  
195 agent to itself i.e. the said application calls another said application where the first said  
196 application is using the second said application and its specific English language memory  
197 systems to get data or cause an action or otherwise respond to the special need of the user  
198 in English; 18) conduct all correspondence to users or other devices in English as either  
199 imperative or declarative English language sentences; 19) dynamically change any or all  
200 sentence memories from external sources – take the existing sentence memories and  
201 switching them with new or different sentences memories; 20) define text and non text  
202 file memories as English language sentences; 21) make a series of sentences that are  
203 being processed to have another set of English Language sentences inserted such that the

204 second set is a subset of the first or inserted somewhere in between the original first set of  
205 sentence and the original last set of sentences; 22) do inference based reasoning where  
206 the inference creates a new English language sentence such that this new sentence  
207 becomes new data for future inferences; 23) store English language sentences such that  
208 the same or similar sentence is stored on top of an existing sentence therefore preserving  
209 the original; 24) provide a command mode such that the said application just responds to  
210 nouns; 25) communicate in English from machine to machine and human to machine  
211 and machine to human; 26) utilize the Darwinian mechanism such that sentences created  
212 by the said application, independent of those created by the user in the storage of English  
213 Language sentences in any of the said applications memories, are created by said  
214 application using inferences so that said inferences (created as English Language  
215 sentences) can be applied against the existing memory system to see if a match can be  
216 found in said memory systems; 27) build English Language sentences by machine that  
217 describe events in the computer and can be stored in text files as memory; 28) accept files  
218 to reprogram itself and or have new functionality; 29) logically unite disparate activities  
219 called from the English Language so that English Language logical analysis can occur to  
220 resolve outcomes and make decisions based on the rules of human language and  
221 governing norms; 30) oversee human language rule patterns based on other systems in  
222 order to analyze those systems and propose new solutions; 31) automatically makes a set  
223 of sentences based on one input sentence so the said application can automatically search  
224 various memories (made up of English Language sentences) to find a target memory that  
225 satisfies the intent of the original sentence; 32) read text stored in text files imported by  
226 any means (video, or any transducer input) so that said application can make judgments

227 of said text by said application based on storage of said application memory (made up of  
228 English Language sentences); 33) learn from other devices such that the said applications  
229 memories in Figure 1 items (5), (8), and (11) are programmed in English Language  
230 sentences from outside devices either by human or machines where machines are defined  
231 as any non human device.

232  
233  
234  
235 Description: Refer to Figure 1. The computer application known as said application (this  
236 computer software application) describes methods that allow a computer to process  
237 English Language sentences of the type: Declarative, Imperative, X1, and X2. Where  
238 input (1) (refer to Figure 1 item (1)) takes in English Language text converted from 1 of  
239 N transducers. Text in ASCII format is fed to rule based parsers in Figure 1 item (2) that  
240 analyze each input sentence and parse that sentence based, in some cases, on a previous  
241 sentence. Said sentence: (What is 3.333 times 2.444?) will display the answer to 2  
242 places when previous sentence to (What is 3.333 times 2.444?) is: (display answer to 2  
243 places.). Said input sentence (What is 3.333. times 2.444?) finds sentence type in Math  
244 memory when user tells the said computer application to: (go to math memory.).  
245 Sequence of said sentences to compute: (What is 3.333 times 2.444?) includes the  
246 following: (Go to math memory. Display answer 2 places. What is 3.333 times 2.444?).  
247 Note: said application will process multiple sentences separated by a period, question  
248 mark or exclamation mark.

250 Said application is configured to open a sentence memory form (8) or (11) memories.

251 The currently open said memory points to a Math memory as defined by a sentence in the

252 current open memory. User (human) or machine inputs sentence: (go to math memory)

253 and instructs the said computer application to switch to Math memory. Said Math

254 memory contains English Language sentences of type math such an input sentence of

255 type: (What is 4 times 4?) matches (Number times Number) in stored memory sentence:

256 (Number times Number is “This will multiply two numbers.”). Said sentence is a

257 declarative sentence containing a noun and prepositional phrase (Number times Number)

258 and a verb phrase: (is “This will multiply two numbers.”) containing the verb: is and the

259 noun phrase contained within the double quotes: (“This will multiply two numbers”).

260 Said sentence verb phrase contains a noun phrase defined with the double quotes even

261 though its literal meaning is not a noun phrase. Said verb phrase is equal to: (is “”).

262 Therefore, said equivalent sentence is: (Number times Number is “”).

263

264 Said application switches to Math memory from said input sentence: (go to math

265 memory.) (or may be done automatically when said application analyzes input sentence

266 and switches between English Language sentence memories), Next sentence in input

267 sentences instructs said computer application to: (display answer to 2 places.). This

268 sentence is found in common memory at said location (5) and implements the said

269 instruction based on the attached action(s) to this said input sentence.

270

271 In last sentence of the above 3 input sentences: (What is 3.333 times 2.444?) matches  
272 said math memory sentence: (Number times Number is “This will multiply two  
273 number.”) and displays the said answer: (The multiplication is 8.15).

274

275 The said computer application applies input sentences from said transducers show in  
276 Figure 1 item (13) to memory systems in Figure 1 items (5), (8), and (11) looking for  
277 fuzzy sentence matches which have attached actions for which any action can be an  
278 English Language sentences with is fed back into the input at Figure 1 item (13). Said  
279 application memory is made up of English Language sentences and said application can  
280 have 1 of N memories as defined by a user or copied into the said application memory  
281 system from other users at Figure 1 items (5), (8), and (11). A memory is called an  
282 object. Said input sentence is directed to 1 of N objects from a previous input sentence  
283 coming from an attached action of that sentence as in Figure 1 items (5), (8), and (11) or  
284 from the input by a user or machine as in Figure 1 item (1). Said input sentence in Figure  
285 1 item (1) or (13) where item (13) is an attached English Language sentence from said  
286 previous input sentence finds a fuzzy sentence match in current open object where said  
287 open object is selected by a previous input English Language sentence. Said open object  
288 containing English Language sentences may be matched by input sentence causing said  
289 object’s sentence to execute said attached action. Said attached action can be the  
290 execution of a computer program, multiple attached English Language sentences or any  
291 program function within or outside of said application. Said computer application can  
292 send and English Language sentence in Figure 1 items (13) and (14) to another computer  
293 running said computer application. In turn, said remote computer application can



294 respond to said computer application English Language sentence by sending back an  
295 English Language sentence to said computer application. Said remote computer  
296 application may decide to send an English Language sentence(s) to other than first said  
297 computer application. Other said computer applications running on other said computers  
298 or other devices may eventually send an English Language sentence back to original said  
299 computer application.

300

301 Said objects (memory containing English Language sentences) of said computer  
302 application can be switched by a said input sentence coming from Figure 1 item(1) or  
303 (13) such that same input can result in different actions depending on said open object.  
304 Said objects can contain same or (fuzzy similar) sentences across all objects but with  
305 different attached actions. Telling said application to: (go to your New York memory.)  
306 and asking: (Who is John Smith?) with said answer: (A person who makes shoes.) vs.  
307 telling said application to: (go to your Flordia memory.) and asking: (Who is John  
308 Smith?) with resulting said answer: (A person who lives at home.). Said application can  
309 send same input sentence (polymorphism) to said objects resulting in different results  
310 based on attachment to same sentence across all objects.

311

312 Said input sentence(s) coming from Figure 1 items (1) or (13) may be used to expose said  
313 objects (memories located in Figure 1 items (5), (8), and (11)) where said object is a class  
314 for said English Language sentences within said object . For example, directing said  
315 computer application to open its vehicle memory obect by telling said computer  
316 application to: (go to vehicle memory.) exposes the object's sentences to the next input

317 sentence so that for example, next said input sentence coming from Figure 1 items (1) or  
318 (13) could be: (What do cars do?). Said object memory (encapsulates) details of vehicles  
319 in said object memory. Once said vehicle object memory is open by telling said  
320 computer application at Figure 1 items (1) or (13) to: (go to vehicle memory.) said next  
321 input sentence at Figure 1 items (1) or (13) is exposed to all of said sentences stored in  
322 said vehicle object memory.

323  
324 Said computer application uses (data abstraction) by telling said computer application at  
325 Figure 1 items (1) and (13) to open its object memory. (where item (1) is input by human  
326 or other devices which compose an English Language sentence or item (13) where item  
327 (13) is an attached sentences stored in Figure 1 items (5), (8), and (11)). Said object  
328 memory is open by input sentence and next input sentence is exposed to said open object  
329 memory. Said open object memory contains all information on the subject of said object  
330 such that any query to said open object memory will contain information relative to the  
331 open object's memory. Said sentence: (go to my vehicle memory.) input at Figure 1  
332 items (1) or (13) exposes next said input sentence to object memories in Figure 1 items  
333 (5), (8), and (11) to a sentence like: (What has 3 wheels?). Where the open memory  
334 vehicle object containing information on all vehicles or attached sentences to other object  
335 memories can respond to a specific requests for all types of vehicles.

336  
337 Said application utilizes (inheritance at the object memory level) when said input  
338 sentence to said computer application in Figure 1 items (1) and (13) causes said object  
339 memory to open in Figure 1 item (8) and (11). Said open object memory contains a said

340 sentence with attached action that causes a common object memory to open in Figure 1  
341 item (5). Said object memory in Figure 1 items (8) and (11) inherits said corresponding  
342 common memory in Figure 1 item (5) so that a base class of sentences is combined with  
343 other object memories and their sentences shown in Figure 1 items (8) and (11). For  
344 example, said input sentence (go to your vehicle memory) in Figure 1 item (1) and (13)  
345 causes said object memory to open in Figure 1 item (8) or (11). Said open object  
346 memory contains a sentence with an attached sentence (Figure 1 item (13)) to open 1 or  
347 N said common memories in Figure 1 item (5). Both object memory sentences from  
348 Figure 1 item (8) or (11) and item (5) are stored in computer RAM. User now inputs next  
349 input sentence at Figure 1 item (1) or (13) to expose open object memories Figure 1 item  
350 (5) and item (8) or Figure 1 item (5) and item (11). The vehicle open object memory in  
351 Figure 1 items (8) and (11), for example, will process input sentences at Figure 1 items  
352 (1) and (13) about vehicle attributes for that open object memory, but it will also process  
353 input sentences regarding 2 and 3 wheeled vehicles based on sentences stored in the  
354 selected common object memory. Therefore, user inputs: (go to your vehicle memory.)  
355 and said computer application opens vehicle memory and then opens the related common  
356 memory that holds sentences and actions regarding 2 and 3 wheeled vehicles. In this  
357 regard the open vehicle memory in Figure 1 item (8) or (11) inherits vehicle base class  
358 information from the corresponding common memory object in Figure 1 item (5).  
359 Typical input sentences would be as follows: (open your vehicle memory. What do cars  
360 do? What has 2 wheels?). Where: (What has 2 wheels?) comes from sentences in the  
361 common object memory (Figure 1 item (5) and opened by the sentence: (What do cars  
362 do?) located in object memories in Figure 1 items (8) and (11).

363

364 Said application utilizes (inheritance at the sentence level) when said application process  
365 English Language sentences to do inference based reasoning. Said example sentences:  
366 (My car will not start. What should I do? vs. What is wrong? For deductive based  
367 reasoning) causes said application to answer: (take a taxi or other derived answer  
368 extracted from existing said application memory). Said application analyzes input  
369 sentences: (My car will not start. What should I do?) and reverses the order so that said  
370 input sentences become: (What should I do? My car will not start.) where said sentence:  
371 (What should I do?) causes common memory in Figure 1 item (5) to set up logic in  
372 Figure 1 item (2) so that said sentence: (My car will not start.) is processed using  
373 inference based reasoning. Attached action of the sentence (My car will not start.) stored  
374 in said application memory in Figure 1 items (5), (8), and (11) is (Vehicles transport  
375 things. Go to vehicle memory.) where said application goes to its vehicle memory  
376 knowing its looking for an inference from the said application based on input sentence:  
377 (What should I do?). Said application process attached sentences: (Vehicles transport  
378 things. Go to vehicle memory.) which goes to input in Figure 1 item (13) from (My car  
379 will not start.) isolating verb (transport) and applying a search in vehicle memory looking  
380 for same said verb (transport). Verb (transport) in said originating sentence with attached  
381 sentence: (Vehicles transport things.) also has a hierarchical number associated with the  
382 sentence: (Vehicles transport things.) of 1. In said target memory: (Go to vehicle  
383 memory.) searched verb (transport) at hierarchical number 1 is found in target memory  
384 with said sentence: (Cars transport people.) with attached inference: (Take a taxi.). Said

385 application displays said inference solution to user (Take a taxi.) based on input  
386 sentences: (My car will not start. What should I do?).

387  
388 The said application has many functions all of which can be implemented by inputting  
389 English Language sentences of the type declarative, imperative, interrogative, and  
390 exclamatory. To put the said application in the learn mode, a sentence like: (please  
391 learn.) and others using fuzzy logic (stored English Language sentences etc.) will store  
392 English language sentences in a declarative format. (Note that fuzzy logic is defined as  
393 storing an English Language declarative (factual) sentence in 1 of N memories shown in  
394 Figure 1 in items (5), (8), and (11) such that the stored (learned sentence) defines like  
395 words by using an English Language thesaurus). All learned sentences will be stored in 1  
396 of N computer memories at locations (5), (8), and (11) in figure 1. Teaching the said  
397 application to learn how to play the computer game of solitaire can be done by storing a  
398 declarative sentence like: (card game of solitaire is “Playing a solitaire card game.”) and  
399 attaching the solitaire program executable (sol.exe) to the said declarative sentence. Any  
400 number of attached actions can be stored with a learned declarative English Language  
401 sentence including other English Language sentences that are fed back into the input at  
402 Figure 1 item (13) or to the output item (14). From the stored sentence: (card game of  
403 solitaire is “Playing a solitaire card game.”) any declarative, imperative, interrogative,  
404 and exclamatory sentence from machine to human can input an ASCII text sentence to  
405 play a solitaire card game using said application. Some of the word combinations  
406 include: (please play cards, get solitaire, play a game, load a card game, Can you play  
407 solitaire?) etc. These word combinations are formed using English Language sentence

rules and implemented by humans or machines and input into said application in Figure 1 item (1) or item (13) to match words in object memory in Figure 1 items (5), (8), and (11). Said input sentences finds object memory sentence in Figure 1 items (5), (8), and (11) which has been learned and stored by user in declarative English Language format as in: (card game of solitaire is “Playing a solitaire card game.”). Each of the input sentences (declarative, imperative, interrogative, and exclamatory) will match various words of the learned/stored declarative English Language sentence to cause said computer application to play the computer game of solitaire. The method described above in Figure 1 item (5) and (8) learns declarative English Language sentences in Native mode and stores those sentences in 1 of N user object memories stored in computer files. File memories are defined by users and contain English Language declarative sentences and associated stored actions as defined by the user. A stored action is any event or another English Language sentence that may be attached to a stored declarative sentence. Said stored actions can instruct said application to switch to a new object memory file for which the next input English Language sentence from said transducers in Figure 1 item (1) can cause new functionality to occur. Likewise said application can use text files (shown in Figure 1 item (11)) that are English Language declarative sentences separated into paragraphs as shown in Figure 2.

1. play a card solitaire game.

do: (sol.exe, c:\Windows).

431 display: Playing the card game of solitaire.

432

433 2. play a board chess game.

434 do: (chess.exe, c:\Windows).

435 display: Looking at my Files.

436

437 3. get my your this text paragraph file.

438 do: (notepad.exe, \Computer text memory\game text memory.txt).

439

440 4. show my computer files folders.

441 do: (Explorer.exe, c:\).

442

443 3x. show what the Ews your architecture diagram lay out layout.

444 do: (c:\Program Files\PowerPointViewer\Ppview32.exe, c:\MyPro386w\PwrPt\diagram3).

445

446

447 3y. let Something destroy the bark.

448 display: going to tree memory.

449 do: go to tree memory.

450

451

452 4. who is Bill William Jefferson Clinton?

453 do: play cards.

454 do: show my computer files.

455 display: A guy who lives in New York.

456 related: His mother lives in Colorado.

457

458

459 5. who is Hillary Hilary Rodam Clinton?

460 display: Wife of the President of the United States.

461 related: Hillary's parents.

462 do: search. play cards.

463

464

465 6. get me picture map of England Great Britain.

466 do: (president.bmp, c:\MyPro386w\bitmap files\uk.bmp).

467 display: Showing a map of England.

468

469

470 7. find a chair car or boat.

471 do: play cards. go to core.

472

473

474 8. show my C disk drive space availability capacity amount on my C drive.

475 do: (Drvspace.exe, c:\).

476 display: Showing the c drive space.



477

478

479 8a. show my A disk drive space availability capacity amount on my C drive.

480 do: (Drvspace.exe, a:\).

481 display: Showing the a drive space.

482

483

484 8b. do disk clean or defragmentation.

485 do: (defrag.exe, c:\Windows).

486 display: defrag the selected disk drive.

487

488

489 9. who owns or is the owner of the Chicago and Northwestern?

490 do: play solitaire.

491

492

493 10. play a chess board game.

494 display: playing chess.

495

496

497 11. call a Person.

498 do: If get Person's phone number. then call Person. else call information.

499 display: sentence logic test. Also, Chuck's phone number should be asserted into

500 memory which would allow call Chuck to happen.

501

502

503 12. call 411 information.

504 do: (sol.exe, c:\Windows).

505

506

507 12a. what does Intel PC notes makes computers easy to use say.

508 do: (Intel ease of use.bmp, c:\MyPro386w\bitmap files\Intel ease of use.bmp).

509 display: From Intel's web site on 12/15/98.

510

511

512 13. He was in New York getting his car.

513 display: getting his car in New York.

514

515

516 14. This is a Chuck test.

517 display: This is a Chuck test.

518

519

520 15. get calculator adder my math machine.

521 do: (calc.exe, c:\Windows).

522 display: can also be made into a scientific calculator.

523

524

525 16. something destroyed the bark.

526 do: play cards.

527

528

529 17. show my design note additions.

530 do: (notepad.exe, a:\other\design notes1).

531 display: These are the design notes on the ews diskette.

532

533

534 18. show the my problems issues list.

535 do: (notepad.exe, a:\Other\ews problem list).

536 display: Ews problem list.

537

538

539 19. go get the other next alternative Bill.

540 do: get test 1. Who is Bill. get test 2.

541 display: Getting information on the other Bill.

542

543

544 20. something destroyed the bark.

545 do: play cards.

546

547

548 21. get the new proposed functions functional capabilities notes list.

549 do: (notepad.exe, a:\Other\function list).

550

551

552 22. set up Ews development environment platforms software tools.

553 do: do Dos. show problem list. show functional notes. show design notes.

554

555

556 Said paragraphs of English Language declarative sentences can be matched by an input

557 English Language sentence from machine to human of type declarative, imperative,

558 interrogative, and exclamatory. Matching the input English Language sentence of type

559 declarative, imperative, interrogative, and exclamatory with English Language

560 declarative sentences stored in each text paragraph as shown in Figure 2 such that one

561 paragraph with attached actions for which one action can be made up of an English

562 Language sentence will call another paragraph sentence who's attached action could call

563 Native memory which switches to a new text file shown in Figure 1 using mechanisms in

564 items (5), (8), and (11). Text file memory switching will enable the following: (Go to

565 Washington, D.C. What is Mr. Smith's phone number? Go to New York. What is Mr.

566 Smith's phone number?). Where the sentences (Go to Washington, D.C. and Go to New

567 York.) switch memories giving the said application through parsers in Figure 1 item (2)

the ability to point to a new set of English Language memories defined in Figure 1 items (5), (8), and (11).

Said application can read a text files made of up of English Language sentences found by telling said computer application to go out and read said text file such that read text file teaches said application by transferring English Language sentences from said text files into said application in Figure 1 item (1). Once said application has transferred English Language sentences into said application memory object in Figure 1 item (8), said application continues to read said text file which further instructs said application, in English Language, to do what said computer application has learned by storing English Language sentences in said computer application object memory in Figure 1 item (8). Said text file that teaches said computer application is shown in Figure 3. Said application is told, using an English Language sentence in Figure 1 item (1) or (13) to learn from a text file when user instructs said computer application to go out and read said text file. The said text file that teaches the said computer application to play the computer game of solitaire is as follows:

Go to your learning memory. Save this data. "sol exe".  
Save the user sentence. "solitaire card game" is "playing solitaire".  
Stop learning text sentences. play solitaire. Go to your user memory.

Figure 3

591

592 Said common memory in Figure 1 item (5) contains English Language declarative  
593 sentences stored in ASCII text such that stored English Language sentence in common  
594 memory is matched with an input English Language input sentence of the form  
595 declarative, imperative, interrogative, and exclamatory located in Figure 1 item(1) or (13)  
596 from machine or human. When said match is made in common object memory with  
597 input sentence, search of subsequent memories in Figure 1 items (8) and (11) is  
598 prevented. If search of common memory fails to find a stored English Language  
599 sentence that matches input sentence using fuzzy logic, search continues in Figure 1 item  
600 (8) and then to item (11). Said application in Native memory in Figure 1 item (8) or item  
601 (11) can have a stored English Language sentence attached to a stored action that  
602 switches common memory in Figure 1 item (6). Said common memory switches to a  
603 new set of English Language sentences which may become aligned to the context of said  
604 Native memory in Figure 1 item (8) and item (11) as defined by the user when said  
605 application is in learn mode storing actions with said memories in Figure 1 item (8) and  
606 (11).

607

608 Said common memory in Figure 1 item (5) enables the following functionality to occur in  
609 said application using deductive and inference based reasoning. In deductive based  
610 reasoning input English Language sentences show in Figure 1 item (1) may include a  
611 stream of declarative sentences such as: (My car will not start. There are no headlights.  
612 What is wrong?). Where said sentence: (What is wrong?) resides in common memory and  
613 causes deductive based processing of said two input sentences: (My car will not start.

614 There are no headlights.). Said common memory sentence stored in Figure 1 item (5)  
615 uses fuzzy logic such that variation of said sentence: (What is wrong?) can include: (get a  
616 solution. What is it?) where these variations are connected to the same action causing  
617 said deductive solution to the English Language input sentences in Figure 1 item (1): (My  
618 car will not start. There are no headlights.). Said common memory sentences (What is  
619 wrong? etc.) have stored actions attached to said sentences as shown in Figure 1 item (5)  
620 causing the appropriate process to occur when said sentences (My car will not start.  
621 There are no headlights.) are processed in memories in Figure 1 items (8) and (11).

622  
623 Said declarative input sentences for deductive based reasoning are fired into computer  
624 memory based on matches found in Native and text memory at locations Figure 1, items  
625 (8) and (11). Stored actions associated with each declarative input sentence define a  
626 confidence factor when all input declarative input sentences match all stored declarative  
627 sentences shown in Figure 1 items (8) and (11). Associated stored actions found with  
628 matching input sentences and stored declarative sentences found in items (8) and (11)  
629 cause an expected result which could include sending a new English Language sentence  
630 to the input located in Figure 1 item (13). The stored actions may contain a number of  
631 sentences the are either fed back to the input in at Figure 1 item (13) or to external  
632 computers, humans, or appliances in Figure 1 item (14). Where an appliance accepts an  
633 English Language sentence, reacts to that sentence, and sends an English Language  
634 sentence back to the sending device as in Figure 1 item (1) or other devices in a network.  
635 All generated English Language sentences resulting from memory transactions whether  
636 generated from a stored action or composed using internal parsing technology in Figure 1

637 item (2) interact with the memory of said computer application or the memories of other  
638 devices creating a dialog using English Language sentences between devices. Any  
639 decoded English Language sentence results in the activation of a stored action. A simple  
640 device may decode specific English Language sentences (decoded to ASCII text for  
641 processing) sent from said application resulting in said device sending back an English  
642 Language sentence in response to said application or other said applications or other  
643 devices or humans either as text, voice or other electrical signals represented by said  
644 English Language sentence(s). Said stored actions result in new sentences or memory  
645 switching as shown in Figure 1 items (4), (6), (7), (9), and (10) from said application or  
646 canned sentences from appliances causing said work to be done for user.

647  
648 When said application is not in the learn mode, English Language declarative input  
649 sentences shown in Figure 1 item (1) find stored declarative sentences in Figure 1 item  
650 (8) and item (11). When found, those declarative sentences are connected to stored  
651 actions which define the solution for deductive based reasoning. Associated solutions  
652 include English Language sentences that may be fed back to the input Figure 1 item (13)  
653 or go outside of the said computer application shown at Figure 1 item (14) to other  
654 devices. Those devices can be said computer application running on other computers or  
655 appliances. In all cases, outside sources, either human or machine, communicate to each  
656 other in English Language sentences.

657  
658 Said computer application can integrate deductive based reasoning with inference based  
659 reasoning. Deductive based reasoning is the storage of English Language sentences



660 connected together within and across object memories. Connecting sentences and object  
661 memory together occurs in the learn mode as user defines connecting relationships  
662 between sentences. User typically enters declarative sentences in Figure 1 items (1) and  
663 (13) which map into learned stored connected sentences in object memory in Figure 1  
664 items (5), (8), and (11). A typically example session could include the following input  
665 sentences: (go to vehicle memory. My car will not start. There is no dome light. What is  
666 wrong?). When said computer application receives first said sentence: (go to vehicle  
667 memory.) said application switches to vehicle object memory in Figure 1 items (8) and  
668 (11). Once switched, said second sentence: (My car will not start.) finds a sentence  
669 match in current open object vehicle memory. Matched said sentence (My car will not  
670 start.) has an attached sentence: (go to start problem memory.) Said computer application  
671 builds new remaining sentences: (go to start problem memory. There is no dome light.  
672 What is wrong?) and said computer application switches to: (go to start problem  
673 memory) in Figure 1 item (13) causing new object memory to open in Figure 1 items (8)  
674 and (11). Next input sentence of remaining sentences: (There is no dome light. What is  
675 wrong?) finds matching sentence in current open object memory (go to start problem  
676 memory.) in Figure 1 items (8) and (11). As said matching sentences are found  
677 throughout object memories, said attached actions to each matching sentence is stored in  
678 said computer application RAM area noting the number of matched sentences compared  
679 to the number of matched sentence with a group as defined by the user in said application  
680 learn mode. Said last sentence: (What is wrong?) causes said application to find said  
681 sentence in object common memory in Figure 1 item (5). Said sentence: (What is  
682 wrong?) has attached actions that causes said application to process data stored in said

683 computer application RAM resulting from said sentence matches found in said object  
684 memory in Figure 1 items (8) and (11). Resulting said solution: (The car battery may be  
685 dead or disconnected. Confidence factor 100 percent). Said data in Ram may also  
686 include sentences: (go to battery memory. battery provides power.) such that when user  
687 inputs: (What should I do? – asking said application to make an inference or alternative  
688 to using a battery, and where said sentence: (What should I do?) exists in said common  
689 object memory in Figure 1 item (5)) said application transfers said sentences: (go to  
690 battery memory. Battery provides power.) to Figure 1 item (13). Said inference  
691 sentences: (go to battery memory. Battery provides power.) switch to battery memory as  
692 directed by input sentence: (go to battery memory.) in Figure 1 items (8) and (11). Next  
693 said input sentence: (Battery provides power.) finds said sentence in battery object  
694 memory in Figure 1 items (8) and (11). Said sentence: (Battery provides power.) has  
695 attached sentence: (go to electrical power memory.) causing said application to switch  
696 object memory in Figure 1 items (8) and (11). Said application isolates verb (provides)  
697 and searches through object memory: (go to electrical power memory.) for said verb  
698 (provides) at same said hierarchical level of 2 as defined by said previous sentence:  
699 (Battery provides power.). Said computer application finds sentence in said open object  
700 memory (go to electrical power memory.) in Figure 1 items (8) and (11) finds sentence:  
701 (Devices provide electrical power.). Said found sentence: (Devices provide electrical  
702 power.) at hierarchical level 2 has attached user comment: (Replace or recharge battery.).

703  
704 Said application uses multi sentence technology to: 1) open new object memory; 2) find  
705 matches in object memory in Figure 1 items (5), (8), and (11) as defined by input from

Figure 1 items (1) and (13); 3) cause said found sentence to execute any action as defined by said attached computer programs; 4) send attached sentences found in memory at Figure 1 items (5), (8), and (11) to output in Figure 1 items (14) to said other computers using said application or devices responding to said English Language sentences or their said symbolic derivatives; 5) enable parallel computing where said computer or human dialog in English Language is sustained using said application within or across computer platforms is complete and dialog stops such that all sentences have competed resulting in work being done by said application. Said found sentences in Figure 1 items (5), (8), and (11) apply new sentences to input at Figure 1 item (13). Input sentences at Figure 1 item (1) or item (13) can be from 1 to N sentences where sentences from Figure 1 item (13) can be attached to stored sentences in Figure 1 items (5), (8), and (11). Input sentences in Figure 1 item (1) can be composed by human or machine in any number from 1 to N.

Said application logically processes input sentences such that the following example input sentences coming from Figure 1 item (1) or (13) from deposited sentences in Figure 1 items (5), (8), and (11) are: (please play cards. If you played cards then show card instructions. Otherwise, play a game of chess.) Where said application finds match in said object memory in Figure 1 items (5), (8) and (11) for said sentence: (please play cards.) noting that if said match did not occur from input in Figure 1 items (1) or (13) and said open object memory in Figure 1 items (5), (8), and (11) that said application generates internal message: (my memory is blank.). When said message is generated (my memory is blank.) because said application could not find match to said input sentence:

729 (please play cards.) to said memory, said next input sentence: (If you played cards then  
730 show card instructions.) is disregarded such that the sentence: (Otherwise, play a game  
731 of chess.) is processed through said application memory in Figure 1 items (5), (8) and  
732 (11) resulting said application playing the computer game of chess. Said application with  
733 said stored English Language sentences stored in said object memories in Figure 1 items  
734 (5), (8), (11) have attached sentences that can be sent to said application output in Figure  
735 1 item (14) or display on said application user screen or put into ASCII text to voice  
736 transducer such that when said input sentences matches said memory sentences with said  
737 attached action which could be a said English Language sentences, that said sentence  
738 could be announced to a local or remote speaker.

739  
740 Said application can be programmed in ordinary English Language sentences by storing  
741 said sentences in said object memories in Figure 1 items (5), (8), and (11). Format of  
742 said stored English Language sentences can be generated by computer program and auto  
743 loaded into said application object memory. Said application can request auto load of  
744 new said application object memories to adapt to new requirements as defined by an  
745 English Language sentence form human or machine.

746  
747 Said application uses two methods to sort through user object memory in Figure 1 items  
748 (5), (8), and (11). Method 1 indexes said object memory in Figure 1 item (8) to open  
749 object memories in Figure 1 item (11) with single sentence input or multi sentence input.  
750 Method 2 builds internal sentences to reapply those sentences in Figure 1 item (13) in  
751 order to sort through object memory in Figure 1 items (5), (8), and (11). In Method 1,

752 human or machine enters sentence(s) in Figure 1 item (1) or said application enters  
753 sentence(s) in Figure 1 item (13). Said object memory switches object memory based on  
754 attached sentence associated with matched sentence in Figure 1 items (5), (8), and (11).  
755 For example, the input sentence(s): (Go to New York memory. Who is John Smith?)  
756 causes said index sentence: (Go to New York memory.) to switch to New York object  
757 memory when said application is told with an English Language sentence to: (please go  
758 into search mode.). (Go to New York memory.) is in an index of all object memories  
759 located in Figure 1 item (8). Attached action of said found sentence (Go to New York  
760 memory.) switches object memory to New York memory in Figure 1 items (8) or item  
761 (11). Said next sentence locates said sentence (Who is John Smith?) through existing  
762 index or in memories in Figure 1 items (11) or items (5). Said open object memory is  
763 open containing 1 of N sentences of which one is fuzzy similar to (Who is John Smith?).  
764 Said next sentence: (Who is John Smith?) searches for a match in said open object  
765 memory in Figure 1 items (8), and (11) and displays answer to (Who is John Smith?) as  
766 defined in said computer application learn mode. In said learn mode, user would have  
767 previously stored and defined a sentence like: (“John Smith New York friend” is “John  
768 Smith is a friend from New York.”) whereby the input sentence: (Who is John Smith?)  
769 in Figure 1 items (1) or (13) would look for a match in Figure 1 items (5), (8), and (11)  
770 such that the input sentence is parsed (broken apart according to the rules: sentence =  
771 noun phrase + verb phrase) in Figure 1 item (2) or their derivatives and applied against  
772 the stored sentence: (Who is John Smith?) found in Figure 1 items (5), (8), and (11).  
773 Found said sentence: (Who is John Smith?) may have 1 of N stored actions. Said stored  
774 actions can include any combination of English Language sentences and other data types

775 depending on user configuration of stored sentence in Figure 1 items (5), (8), and (11).  
776 Said stored actions are attached to said stored sentence in said computer application using  
777 said computer application learn mode. Said stored sentences and said stored actions are  
778 stored in Native, text or SQL data types as shown in Figure 1 items (5), (8), and (11).